EXPRESS MAIL NO. EL897867380US

MODULARIO LCA - 101

Mod. C.E. - 1-4-7 #4

Ministero delle Attività Produttive

Direzione Generale per lo Sviluppo Produttivo e la Competitività
Ufficio Italiano Brevetti e Marchi
Ufficio G2

Invenzione Industriale

Autenticazione di copia di documenti relativi alla domanda di brevetto per:

N. TO2000 A 001049

Si dichiara che l'unita copia è conforme ai documenti originali depositati con la domanda di brevetto sopraspecificata, i cui dati risultano dall'accluso processo verbale di deposito.

Roma, li 28 GEN. 2002

IL DIRIGENTE

Caso 00-AG-088
Ns.Rf.2/3751

AL MINISTERO DELL'INDUSTRIA DEL COMMERCIO E DELL'ARTIGIANATO **MODULO A**
UFFICIO ITALIANO BREVETTI E MARCHI - ROMA
DOMANDA DI BREVETTO PER INVENZIONE INDUSTRIALE, DEPOSITO RISERVE, ANTICIPATA ACCESSIBILITÀ AL PUBBLICO

marca da bollo

A. RICHIEDENTE (I)
1) Denominazione: STMICROELECTRONICS S.R.L. — N.G.: SR
Residenza: AGRATE BRIANZA (MI) — codice: 00951900968
2) Denominazione:
Residenza: — codice:

B. RAPPRESENTANTE DEL RICHIEDENTE PRESSO L'U.I.B.M.
cognome e nome: BERGADANO MIRKO e altri — cod. fiscale:
denominazione studio di appartenenza: STUDIO TORTA S.r.l.
via: Viotti — n.: 0009 — città: TORINO — cap: 10121 — (prov): TO

C. DOMICILIO ELETTIVO destinatario
via: — n.: — città: — cap: — (prov):

D. TITOLO — classe proposta (sez/cl/scl): — gruppo/sottogruppo:
METODO DI COSTRUZIONE DI UN CODICE A CONTROLLO DELL'ERRORE POLIVALENTE PER CELLE DI MEMORIA MULTILIVELLO FUNZIONANTI A UN NUMERO VARIABILE DI LIVELLI DI MEMORIZZAZIONE E METODO POLIVALENTE DI CONTROLLO DELL'ERRORE UTILIZZANTE TALE CODICE A CONTROLLO DELL'ERRORE.

ANTICIPATA ACCESSIBILITÀ AL PUBBLICO: SI ☐ NO ☐ — **SE ISTANZA: DATA** ☐☐/☐☐/☐☐ — **N° PROTOCOLLO**

E. INVENTORI DESIGNATI (cognome nome)
1) GREGORI Stefano
2) FERRARI Pietro
3) TORELLI Guido
4)

F. PRIORITÀ

	nazione o organizzazione	tipo di priorità	numero di domanda	data di deposito	allegato S/R
1)				☐☐/☐☐/☐☐☐☐	☐
2)				☐☐/☐☐/☐☐☐☐	☐

SCIOGLIMENTO R. Data: ☐☐ ☐☐ ☐☐ ☐☐☐☐☐☐☐☐

G. CENTRO ABILITATO DI RACCOLTA COLTURE DI MICRORGANISMI, denominazione:

H. ANNOTAZIONI SPECIALI
Per la migliore comprensione dell'invenzione è stato necessario depositare disegni con diciture come convenuto dalla Convenzione Europea sulle formalità alle quali l'Italia ha aderito.

DOCUMENTAZIONE ALLEGATA

	N. es.			
Doc. 1)	2	PROV	n. pag. 52	riassunto con disegno principale, descrizione e rivendicazioni (obbligatorio 1 esemplare) .....
Doc. 2)	2	PROV	n. tav. 06	disegno (obbligatorio se citato in descrizione, 1 esemplare .....
Doc. 3)	1	RIS		lettera d'incarico, procura o riferimento procura generale .....
Doc. 4)	1	RIS		designazione inventore .....
Doc. 5)	☐	RIS		documenti di priorità con traduzione in italiano .....
Doc. 6)	☐	RIS		autorizzazione o atto di cessione .....
Doc. 7)	☐			nominativo completo del richiedente

SCIOGLIMENTO RISERVE — Data / N° Protocollo
☐☐/☐☐/☐☐/☐☐☐☐☐☐ — conferma singola priorità ☐☐/☐☐/☐☐/☐☐☐☐☐☐

8) attestati di versamento, totale lire: Novecentoquindicimila= — obbligatorio

COMPILATO IL 07/11/2000 — **FIRMA DEL (I) RICHIEDENTE (I)**
CONTINUA SUINO NO — BERGADANO MIRKO

DEL PRESENTE ATTO SI RICHIEDE COPIA AUTENTICA SUINO SI

CAMERA DI COMMERCIO IND. ART. AGR. DI TORINO — codice 01
VERBALE DI DEPOSITO — **NUMERO DI DOMANDA** TO 2000A 001049 — Reg.
L'anno duemila, il giorno sette, del mese di Novembre
Il (I) richiedente (I) sopraindicato (I) ha (hanno) presentato a me sottoscritto la presente domanda, corredata di n. ☐ fogli aggiuntivi per la concessione del brevetto soprariportato.

I. ANNOTAZIONI VARIE DELL'UFFICIO ROGANTE

IL DEPOSITANTE
STUDIO TORTA s.r.l.
Andrea CROVERI

C.C.I.A.A. Torino

L'UFFICIALE ROGANTE
Loredana ZELLADA

Caso 00-AG-088
Ns.Rf.2/3751

PROSPETTO A

RIASSUNTO INVENZIONE CON DISEGNO PRINCIPALE

NUMERO DOMANDA TO 2000 A 001049 **REG. A**

DATA DI DEPOSITO 07/11/2000

NUMERO BREVETTO _____ **DATA DI RILASCIO** __/__/____

A. RICHIEDENTE (I)

Denominazione STMICROELECTRONICS S.R.L.

Residenza AGRATE BRIANZA (MI)

D. TITOLO

METODO DI COSTRUZIONE DI UN CODICE A CONTROLLO DELL'ERRORE POLIVALENTE PER CELLE DI MEMORIA MULTILIVELLO FUNZIONANTI A UN NUMERO VARIABILE DI LIVELLI DI MEMORIZZAZIONE E METODO POLIVALENTE DI CONTROLLO DELL'ERRORE UTILIZZANTE TALE CODICE A CONTROLLO DELL'ERRORE.

Classe proposta (sez/cl/scl) ____ **(gruppo/sottogruppo)** ____/____

L. RIASSUNTO

Viene descritto un metodo di costruzione di un codice a controllo dell'errore polivalente per celle di memoria multilivello funzionanti a un numero variabile di livelli di memorizzazione, in particolare celle di memoria i cui livelli di memorizzazione possono assumere i valori dell'insieme $\{b^{a_1}, b^{a_1 a_2}, \ldots, b^{a_1 a_2 \cdots a_h}\}$, con b, a_1, ..., a_h interi positivi; il codice a controllo dell'errore codificando parole di informazione (i), formate da k simboli q-ari, cioè appartenenti a un alfabeto contenente q simboli distinti, con $q \in \{b^{a_1}, b^{a_1 a_2}, \ldots, b^{a_1 a_2 \cdots a_h}\}$, in corrispondenti parole di codice (c) formate da n simboli q-ari, con $q=b^{a_1 a_2 \cdots a_h}$, ed avente una capacità di correzione dell'errore t, ogni parola di codice (c) essendo generata attraverso una operazione di moltiplicazione fra la corrispondente parola di informazione (i) ed una matrice generatrice (G). Il metodo di costruzione comprende le fasi di: acquisire i valori di k, t, b^{a_1}, $b^{a_1 a_2}$, ..., $b^{a_1 a_2 \cdots a_h}$, che costituiscono le specifiche di progetto di detto codice a controllo dell'errore; calcolare, in funzione di $q=b^{a_1}$, k e t, il minimo valore di n tale che sia soddisfatto il limite di Hamming; calcolare i valori massimi $\hat{n}$ e $\hat{k}$ rispettivamente di n e di k che soddisfano il limite di Hamming per $q=b^{a_1}$, t e $(\hat{n}-\hat{k})=(n-k)$; determinare, in funzione di t, la matrice generatrice (G_1) del codice a controllo dell'errore abbreviato $(n-k)$ sul campo a elementi finiti $GF(b^{a_1})$; costruire rappresentazioni polinomiali binarie dei campi a elementi finiti $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_h})$; identificare, utilizzando le suddette rappresentazioni esponenziali, gli elementi del campo a elementi finiti $GF(b^{a_1 a_2 \cdots a_h})$ isomorfi agli elementi dei campi a elementi finiti $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_{h-1}})$; stabilire corrispondenze biunivoche fra gli elementi dei campi a elementi finiti $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_{h-1}})$ e gli elementi del campo a elementi finiti $GF(b^{a_1 a_2 \cdots a_h})$ ad essi isomorfi; e sostituire ciascuno degli elementi della detta matrice generatrice (G_1) con il corrispondente elemento isomorfo del campo a elementi finiti $GF(b^{a_1 a_2 \cdots a_h})$, ottenendo così una matrice generatrice polivalente (G_2) definente, assieme alle suddette corrispondenze biunivoche, un codice a controllo dell'errore polivalente utilizzabile con celle di memoria i cui livelli di memorizzazione possono assumere i valori dell'insieme $\{b^{a_1}, b^{a_1 a_2}, \ldots, b^{a_1 a_2 \cdots a_h}\}$.

M. DISEGNO

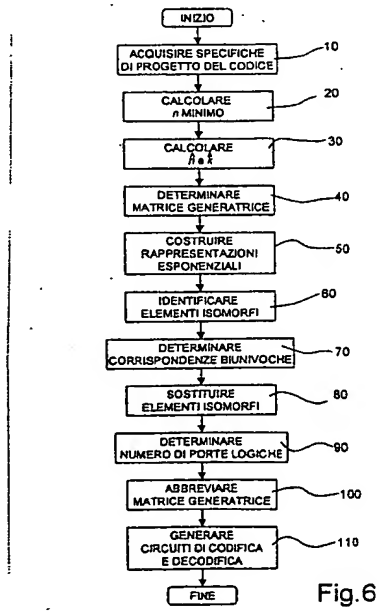
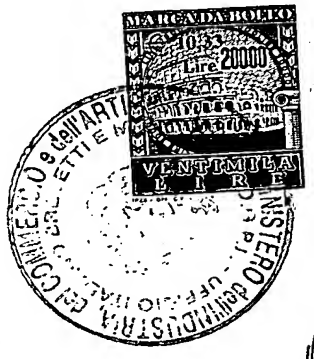


Fig.6



D E S C R I Z I O N E

del brevetto per invenzione industriale

di STMICROELECTRONICS S.R.L.

di nazionalità italiana,

con sede a 20041 AGRATE BRIANZA (MILANO) - VIA C. OLIVETTI, 2

Inventori: GREGORI Stefano, FERRARI Pietro, TORELLI Guido

*** **** ***TO 2000A 001049

La presente invenzione è relativa a un metodo di costruzione di un codice a controllo dell'errore polivalente per celle di memoria multilivello funzionanti a un numero variabile di livelli di memorizzazione e a un metodo polivalente di controllo dell'errore utilizzante tale codice a controllo dell'errore.

In particolare, la presente invenzione riguarda la definizione di codici a controllo dell'errore per memorie a semiconduttore multilivello e, più precisamente, di codici lineari a blocco polivalenti che permettono la rivelazione e la correzione dell'errore in memorie multilivello mantenendo la loro funzionalità con celle di memoria operanti a un differente numero di livelli di memorizzazione.

Come è noto, grazie all'evoluzione dei processi tecnologici che rende realizzabili dispositivi elementari di dimensioni sempre più ridotte, negli

ultimi anni sono state realizzate memorie a semiconduttore aventi capacità di memorizzazione molto elevate.

Un ulteriore aumento della capacità di memorizzazione è stato ottenuto ricorrendo alla memorizzazione multilivello, la quale permette di aumentare la densità di memorizzazione a parità di generazione tecnologica. Infatti, con questa tecnica si memorizzano più bit d'informazione all'interno della singola cella di memoria normalmente utilizzata per contenere un solo bit.

È altresì noto che per leggere una cella di memoria bilivello (contenente 1 bit) si confronta un'opportuna grandezza elettrica, legata allo stato della cella, con un valore di riferimento e in base all'esito del confronto si determina se la cella di memoria contiene uno "0" oppure un "1" logico.

Nel caso di celle in grado di contenere r bit, la lettura avviene confrontando la grandezza elettrica correlata allo stato della cella con 2^r-1 livelli di riferimentó. L'esito dei confronti permette di determinare in quale dei 2^r intervalli ammessi si trova la cella, e quindi di ricostruirne il contenuto in termini di informazione binaria.

L'approccio multilivello può essere applicato sia

alle memorie volatili (come le memorie DRAM) sia alle memorie nonvolatili (come le memorie EEPROM e Flash). In ogni caso l'aumento del numero di bit per cella rende più critica la tolleranza ai disturbi, la ritenzione dell'informazione e l'accuratezza delle operazioni di lettura e di scrittura. Inoltre, l'incremento della capacità di memorizzazione richiesto dal mercato tende a ridurre l'affidabilità complessiva. Per questi motivi si prevede che l'utilizzo di codici a controllo dell'errore sarà fondamentale soprattutto per memorie multilivello a elevata capacità.

Al momento, i dispositivi commerciali a maggior capacità contengono alcune centinaia di milioni di bit, e nei prossimi anni è prevista la realizzazione di dispositivi con capacità via via più elevata.

L'aumento del numero di celle tende a ridurre la vita media al guasto (o MTTF) dell'intero dispositivo di memoria. Ma data l'esigenza di realizzare apparecchiature o sistemi sempre più affidabili, il livello di affidabilità richiesto per il singolo componente di memoria diventa sempre più stringente. Per questo motivo si adottano tecniche di progettazione dedicate e un attento controllo di qualità sui processi produttivi per prevenire e ridurre i guasti.

Tuttavia i malfunzionamenti dei chip di memoria non

possono essere eliminati completamente e possono essere ridotti solo a spese di una riduzione delle prestazioni o di un aumento dei costi.

Un modo molto efficace per aumentare l'affidabilità è costituito dalla progettazione di memorie immuni dall'errore utilizzando codici a controllo dell'errore, ossia codici in grado di rivelare e correggere errori dei dati memorizzati nelle memorie.

In particolare codici a correzione del singolo errore o a rivelazione del doppio errore e a correzione del singolo errore sono utilizzati in dispositivi di memoria a semiconduttore di vario tipo. A tale proposito si veda ad esempio K. Furutani, K. Arimoto, H. Miyamoto, T. Kobayashi, K.-I. Yasuda, and K. Mashiko, "*A Built-in Hamming Code ECC Circuit for DRAM's*", IEEE J. Solid-State Circuits, vol. 24, no. 1, Feb. 1989, pp. 50-56, e T. Tanzawa, T. Tanaka, K. Takeuchi, R. Shirota, S. Aritome, H. Watanabe, G. Hemink, K. Shimizu, S. Sato, Y. Takeuchi, K. Ohuchi, "*A compact on-chip ECC for low cost flash memories*", IEEE J. Solid-State Circuits, vol. 32, no. 5, Maggio 1997, pp. 662-669.

Gli errori nelle memorie sono normalmente classificati come errori "soft" ed errori "hard". Con errore "soft" si intende un cambiamento dello stato di una cella casuale, non ripetitivo e non permanente. Gli

errori "soft" sono causati da rumore elettrico occasionale o indotti da radiazione (particelle α, raggi cosmici, ...), riguardano un numero molto limitato di celle per volta e sono recuperabili col successivo ciclo di scrittura.

Con errore "hard" si intende invece un guasto fisico permanente associato a difetti presenti nel dispositivo o creatisi durante il suo funzionamento per incapacità dei materiali di sopportare gli stress applicati. Generalmente gli errori "hard" sono molto più rari degli errori "soft".

I codici a controllo dell'errore permettono di ridurre drasticamente gli effetti degli errori "soft", che rappresentano il problema più grave, specialmente per le memorie multilivello. Essi possono peraltro risultare utili anche al fine di recuperare alcuni errori "hard".

Per proteggere l'informazione da immagazzinare nella memoria è necessario aggiungere ai bit che costituiscono ogni parola di informazione un certo numero di bit di controllo, opportunamente calcolati. L'operazione che associa a ogni parola di informazione un preciso valore dei bit di controllo viene chiamata *codifica*. I bit di controllo calcolati dal circuito che effettua la codifica devono essere memorizzati

unitamente alla parola di informazione.

Ogni parola memorizzata sarà successivamente letta insieme con i bit di controllo che le competono. Il circuito di *decodifica* è in grado di rivelare e correggere un certo numero di bit errati per parola confrontando opportunamente il valore dei bit di controllo con il valore dei bit di informazione.

Il numero di bit di controllo che è necessario aggiungere a ogni parola di informazione viene determinato in base alla lunghezza della parola di informazione stessa ed al numero di errori per parola che si vogliono correggere.

Più in generale, la codifica a controllo dell'errore può essere estesa dall'alfabeto binario (contenente solo i due simboli "0" e "1") a un alfabeto più esteso contenente q simboli. In questo caso la codifica consiste nell'aggiunta di un certo numero di simboli (non più di bit) a ogni parola da memorizzare, e la correzione degli errori consiste nella correzione dei simboli errati.

Questa estensione al caso q-ario si addice in modo particolare alle memorie multilivello, in cui ogni cella di memoria è in grado di immagazzinare più di un bit (ad esempio r bit). In questo caso, infatti, il malfunzionamento di una cella di memoria può degradare

il valore di tutti i bit in essa memorizzati. Risulta quindi più agevole associare a ogni blocco di r bit, memorizzati in unica cella, un simbolo q-ario, cioè appartenente a un alfabeto costituito da $q=2^r$ simboli distinti. Ogni simbolo viene quindi memorizzato in una cella di memoria multilivello distinta. In questo modo, ogni parola di informazione di k bit viene vista come una parola costituita da k/r simboli q-ari (pari al numero di celle di memoria che formano ogni parola), e la correzione di un simbolo equivale alla correzione di tutti gli r bit a esso associati.

I metodi di controllo dell'errore integrati in una memoria a semiconduttore devono soddisfare tre requisiti fondamentali:

- il tempo richiesto per l'operazione di codifica e per l'operazione di decodifica (comprendente la rivelazione e la correzione dell'errore) deve influire solo in minima parte sul tempo d'accesso alla memoria;

- l'area aggiuntiva dovuta ai circuiti di codifica e decodifica e alle celle di controllo deve essere minimizzata;

- la tecnica usata deve almeno garantire la correzione di qualsiasi tipo di errore sulla singola cella, che nel caso di celle multilivello può consistere nell'errore su più bit.

Affinché i tempi di codifica e decodifica non degradino il tempo di accesso, si ricorre tipicamente all'utilizzo di strutture di codifica parallela o a matrice, che offrono le maggiori velocità di calcolo. Per una trattazione più dettagliata sull'argomento si veda ad esempio C. V. Srinivasan, "Codes for error correction in high-speed memory systems - part I: correction of cell defects in integrated memories" *IEEE Trans. Comput.*, vol. C-20, no. 8, Agosto 1971, pp. 882-888.

Per quanto riguarda invece il secondo punto, l'area viene minimizzata scegliendo codici con caratteristiche adatte all'applicazione ed opportunamente ottimizzati.

L'ultimo punto è infine garantito dall'utilizzo di codici q-ari, che consentono la rivelazione e la correzione degli errori sulle celle di memoria, indipendentemente dal numero di bit errati associati a ciascuna di esse.

Le memorie multilivello progettate per contenere r bit per cella possono però anche funzionare immagazzinando un numero minore di bit per cella. In questo caso per la scrittura e per la lettura si può usare un sottoinsieme dei 2^r-1 livelli di riferimento disponibili. L'esempio estremo (e più semplice) di questa condizione di funzionamento si ha quando si

utilizza una memoria multilivello come normale memoria bilivello.

La scelta di diminuire il numero di livelli riduce la capacità di memorizzazione della memoria, ma ne aumenta l'affidabilità. Ad esempio, nel caso delle memorie nonvolatili, la riduzione del numero dei livelli consente di garantire la ritenzione dell'informazione per un tempo più lungo e in condizioni ambientali più sfavorevoli.

Normalmente la scelta della modalità di funzionamento viene effettuata in modo permanente dal produttore; in questo caso la possibilità di cui sopra può essere interessante ad esempio per ottenere una memoria con un numero minore di bit per cella come sottoselezione di una progettata per contenere un numero maggiore di bit, al fine di una complessiva riduzione dei costi.

Attualmente, però, per soddisfare le crescenti richieste del mercato, si stanno progettando dispositivi di memoria in cui anche l'utilizzatore finale possa decidere la modalità di funzionamento in base al tipo di impiego del dispositivo, e di conseguenza è sempre più sentita l'esigenza di realizzare un metodo di controllo dell'errore polivalente che sia in grado, utilizzando gli stessi circuiti, di proteggere i dati immagazzinati

in celle che funzionano a un numero diverso di livelli.

Tale esigenza è ulteriormente rafforzata dal fatto che i dispositivi di memoria delle prossime generazioni, con grande capacità di memorizzazione, potranno essere configurabili settore per settore ed avere quindi settori interni impostati con un numero diverso di bit per cella (si veda ad esempio il brevetto statunitense US 5,574,879).

Dispositivi di memoria di questo tipo potranno ad esempio essere utilizzati all'interno di carte multimediali, consentendo di memorizzare in settori a basso numero di bit per cella il microcodice per il microprocessore che gestisce la carta e in settori ad alto numero di bit per cella i dati dell'utente.

Scopo della presente invenzione è quindi quello di fornire un metodo di costruzione di un codice a controllo dell'errore per celle di memoria multilivello funzionanti a un numero variabile di livelli di memorizzazione ed un metodo polivalente di controllo dell'errore utilizzante tale codice a controllo dell'errore che consentano, utilizzando gli stessi circuiti, di proteggere i dati immagazzinati in celle che funzionano a un numero diverso di livelli.

Secondo la presente invenzione viene fornito un metodo di costruzione di un codice a controllo

dell'errore per celle di memoria multilivello funzionanti a un numero variabile di livelli di memorizzazione, come definito nella rivendicazione 1.

Secondo la presente invenzione viene fornito un metodo polivalente di controllo dell'errore per celle di memoria multilivello funzionanti a un numero variabile di livelli di memorizzazione, come definito nella rivendicazione 10.

Per una migliore comprensione della presente invenzione viene ora descritta una forma di realizzazione preferita, a puro titolo di esempio non limitativo e con riferimento ai disegni allegati, nei quali:

- le figure 1, 2 e 3 mostrano tabelle relative al metodo di costruzione secondo la presente invenzione;

- le figure 4a, 4b e 5a, 5b mostrano circuiti moltiplicatori utilizzati per l'implementazione del metodo di controllo dell'errore secondo la presente invenzione;

- la figura 6 mostra un diagramma di flusso delle operazioni relative alla costruzione del metodo di controllo dell'errore secondo la presente invenzione;

- la figura 7 mostra un diagramma a blocchi rappresentativo delle operazioni di codifica, memorizzazione e decodifica di una parola di

informazione relative a un codice a controllo dell'errore non polivalente secondo l'arte nota; e

- la figura 8 mostra un diagramma a blocchi rappresentativo delle operazioni di codifica, memorizzazione e decodifica di una parola di informazione relative a un codice a controllo dell'errore polivalente secondo la presente invenzione.

Per facilitare la comprensione della presente invenzione qui di seguito vengono introdotte alcune notazioni relative alla codifica lineare a blocco utilizzata dalla presente invenzione per rivelare e correggere errori in memorie multilivello. Per una trattazione più dettagliata di tale argomento si vedano ad esempio W. W. Peterson, E. J. Weldon, *Error-Correcting Codes*, 2nd ed., M.I.T. Press, Cambridge (Massachusetts), 1972, ed R. E. Blahut, *Theory and Practice of Error Control Codes*, Addison-Wesley, Reading (Massachusetts), 1983.

In particolare, chiamiamo simboli q-ari gli elementi di un alfabeto contenente q simboli distinti, chiamiamo *parola di informazione* il vettore di k simboli q-ari da scrivere nella memoria e lo indichiamo con:

$$\underline{i}=(i_1,\ i_2,\ \ldots,\ i_k)$$

Nella codifica lineare a blocco, la parola di

informazione viene mappata biunivocamente in un vettore di n simboli q-ari (con $n>k$), che chiamiamo *parola di codice* e indichiamo con:

$$\underline{c}=(c_1,\ c_2\ldots,\ c_n)$$

L'operazione di codifica può essere descritta in forma algebrica introducendo la matrice **G**, chiamata *matrice generatrice* del codice (si veda a tale proposito W. W. Peterson, E.J. Weldon, *Error-Correcting Codes*, 2nd ed., M.I.T. Press, Cambridge (Massachusetts), 1983).

Ogni parola di codice $\underline{c}$ può essere generata semplicemente effettuando il prodotto della parola di informazione per la matrice generatrice, come espresso nella seguente equazione:

$$\underline{c}=\underline{i}\cdot G$$

dove $\underline{i}$ e $\underline{c}$ sono vettori riga.

La matrice **G** è una matrice rettangolare con k righe, n colonne e rango k (affinché l'applicazione $\underline{i} \rightarrow \underline{c}$ sia iniettiva).

Per ogni codice lineare (n, k) esiste una matrice **H**, chiamata *matrice del controllo di parità* (si veda il succitato testo *Error-Correcting Codes*), avente $n-k$ righe e n colonne, tale che:

$$\mathbf{G} \cdot {}^{t}\mathbf{H} = \mathbf{0}$$

dove $\mathbf{0}$ indica la matrice $k \times (n-k)$ con elementi tutti nulli, e ${}^{t}\mathbf{H}$ indica la trasposta della matrice $\mathbf{H}$.

Gli $n-k$ simboli, aggiunti durante la codifica, prendono il nome di *simboli di parità* o *simboli di controllo*.

Chiamiamo *codice* q-ario (n, k) l'insieme delle q^k parole di codice ottenute dalla codifica di tutte le possibili parole di informazione e definiamo *capacità di correzione* del codice q-ario (n, k) il numero t di errori (cioè di simboli q-ari errati) per parola che il codice è in grado di correggere.

La seguente disequazione, nota col nome di limite di Hamming (si veda il succitato testo *Error-Correcting Codes*), fornisce il minimo numero di simboli di parità richiesti affinché un codice lineare q-ario (n, k) abbia capacità di correzione t:

$$q^{n-k} \geq \sum_{i=0}^{t} \binom{n}{i} (q-1)^i \qquad (1)$$

Alla luce delle notazioni sopra introdotte, verrà qui di seguito descritta la procedura di costruzione della matrice generatrice di un codice b^s-ario (n, k), tale che la stessa matrice possa essere utilizzata anche per effettuare le operazioni di codifica e decodifica

per un codice b-ario (n, k) avente la stessa capacità di correzione del codice di partenza. In questo modo, si ottiene un unico circuito di codifica e un unico circuito di decodifica utilizzabili per entrambi i codici.

In particolare, al solo scopo di facilitare la comprensione del problema della costruzione della matrice generatrice, verrà ora illustrata l'applicazione del metodo secondo la presente invenzione a un esempio specifico, che comunque non limita l'applicabilità al caso generale, e successivamente verrà dimostrata in termini rigorosi la fattibilità di quanto illustrato nell'esempio specifico.

Si consideri ad esempio il caso di una memoria a 16 livelli che possa essere utilizzata anche come memoria a 4 livelli. Si vuole costruire un circuito per la correzione dell'errore in grado di correggere errori che coinvolgono gruppi di 4 bit associati alla stessa cella di memoria. Allo stesso tempo il circuito deve essere in grado di effettuare la correzione anche nella modalità a 4 livelli, cioè di correggere errori che coinvolgono gruppi di 2 bit associati alla stessa cella di memoria.

Si supponga che per l'applicazione a cui è destinato il codice sia sufficiente correggere un solo errore per parola (t=1) e che ogni parola di

informazione, nella modalità di funzionamento a 16 livelli, sia costituita da 16 simboli esadecimali (k=16, b^s=16), mentre nella modalità a 4 livelli sia costituita da 16 simboli quaternari (k=16, b=4).

Innanzitutto si deve valutare il numero di simboli di parità richiesti in entrambe le modalità di funzionamento, ricorrendo al limite di Hamming. In questo caso (k=16, t=1), grazie alla relazione (1) si ricava che sia in base 16 (q=16), sia in base 4 (q=4), sono richiesti 3 simboli di parità (quindi n=19). La matrice generatrice del codice in base 16 utilizzabile anche come matrice generatrice del codice in base 4 può essere ottenuta nel modo seguente.

Per prima cosa si costruisce la matrice generatrice del codice in base 4 che soddisfa i requisiti iniziali per i valori di n, k e t. Nell'esempio considerato, una matrice quaternaria che soddisfa i requisiti specificati può essere ottenuta abbreviando il codice di Hamming (21, 18) in base 4 (per le definizioni di codice di Hamming e di operazione di abbreviazione si veda il succitato *Theory and Practice of Error Control Codes*). Un esempio è dato dalla seguente matrice:

$$G_1 = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 3 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 2 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 3 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 3 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 2 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 2 & 2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 2 & 3 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 3 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 3 & 1
\end{pmatrix}$$

Si vuole ora convertire la matrice quaternaria G_1 in una matrice esadecimale, che generi un codice in base 16 con la stessa capacità di correzione del codice quaternario di partenza (riferita però a simboli esadecimali).

Si tratta di trovare la giusta corrispondenza biunivoca tra gli elementi quaternari {0, 1, 2, 3} e quattro degli elementi esadecimali {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}. I quattro elementi esadecimali corrispondenti agli elementi quaternari {0, 1, 2, 3} sono detti elementi *isomorfi* agli elementi quaternari {0, 1, 2, 3}.

La corrispondenza deve essere tale da fornire una matrice che, quando viene utilizzata in modalità a 4 livelli, generi parole di codice che contengano solo i 4 simboli esadecimali scelti.

Nel seguito verrà descritto come ottenere le

corrispondenze che soddisfano questa condizione per il caso generale.

Nel caso specifico preso ad esempio esistono due corrispondenze biunivoche che permettono una corretta conversione della matrice generatrice G_1:

$0 \leftrightarrow 0$	$0 \leftrightarrow 0$
$1 \leftrightarrow 0$	$1 \leftrightarrow 1$
$2 \leftrightarrow 6$	$2 \leftrightarrow B$
$3 \leftrightarrow 7$	$3 \leftrightarrow A$

In atre parole, secondo la prima corrispondenza gli elementi esadecimali isomorfi agli elementi quaternari {0, 1, 2, 3} sono rispettivamente {0, 1, 6, 7}, mentre secondo la seconda corrispondenza gli elementi esadecimali isomorfi agli elementi quaternari {0, 1, 2, 3} sono rispettivamente {0, 1, A, B}.

Utilizzando la prima corrispondenza, come matrice generatrice si ottiene:

$$G_2 = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 6 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 7 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 6 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 7 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 6 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 7 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 6 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 6 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 6 & 6 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 6 & 7 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 7 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 7 & 1
\end{pmatrix}$$

Visto che i circuiti digitali che effettuano la codifica e la decodifica funzionano in logica binaria, è necessario descrivere le operazioni che interessano la matrice G_2 in termini di operazioni binarie. Inoltre gli elementi esadecimali devono essere trattati sfruttando la loro notazione binaria.

La matrice G_2 così ottenuta è in grado di funzionare in modalità 4 e 16 livelli, posto che in modalità 4 livelli i simboli quaternari (ovvero le coppie di bit) vengano mappati in simboli esadecimali (ovvero in gruppi di 4 bit) seguendo la stessa corrispondenza biunivoca utilizzata per ottenere la matrice. In questo caso:

$$00\ (=0) \leftrightarrow 0000\ (=0)$$

$$01\ (=1) \leftrightarrow 0001\ (=1)$$

$$10\ (=2) \leftrightarrow 0110\ (=6)$$

$$11\ (=3) \leftrightarrow 0111\ (=7)$$

I circuiti di codifica e decodifica richiedono sempre lo stesso numero di bit di ingresso e forniscono sempre lo stesso numero di bit di uscita, indipendentemente dal fatto che la modalità di funzionamento sia a 16 o a 4 livelli. Nel caso di funzionamento della memoria a 4 livelli è compito di particolari circuiti che precedono il codificatore e che seguono il decodificatore quello di mappare ogni simbolo

quaternario (costituito da 2 bit) nel corrispondente simbolo esadecimale (costituito da 4 bit). In particolare, la rete che precede il codificatore trasforma gruppi di 2 bit in gruppi di 4 bit secondo la corrispondenza sopra riportata, mentre la rete che segue i decodificatore opera la trasformazione di gruppi di 4 bit in gruppi di 2 bit sempre secondo la stessa corrispondenza. Una spiegazione più dettagliata di questo aspetto dell'invenzione verrà effettuata in seguito con riferimento alle figure 7 e 8.

Si intende ora dimostrare in termini rigorosi la fattibilità di quanto appena esposto relativamente al succitato esempio specifico.

In particolare, la dimostrazione che segue verrà effettuata prendendo in considerazione la base binaria, ossia verrà spiegato come ottenere una relazione biunivoca che consente la conversione di elementi b-ari in elementi b^s-ari, con b=2.

Quanto detto relativamente al caso b=2, tuttavia, è applicabile tale e quale alla conversione di elementi b-ari in elementi b^s-ari, con b qualsiasi.

Per effettuare questa dimostrazione rigorosa è indispensabile richiamare alcune nozioni di algebra dei campi finiti.

Indichiamo con $GF(q^s)$ il *campo di Galois* (cioè il

campo a elementi finiti) contenente q^s elementi (per una trattazione dettaglia dell'argomento si veda il succitato *Theory and Practice of Error Control Codes*).

Siccome $GF(q^s)$ è un campo finito, moltiplicando un suo elemento ß per se stesso un certo numero di volte si ottiene ancora ß. Si definisce *ordine* di ß il minimo intero i per cui $ß^i=ß$ (per una trattazione dettaglia dell'argomento si veda il succitato *Error-Correcting Codes*).

Dati due campi $GF(q^s)$ e $GF(q^r)$, si può dimostrare che $GF(q^r)$ è un *sottocampo* di $GF(q^s)$ se e solo se s è divisibile per r. In tal caso $GF(q^s)$ è detto *campo esteso* di $GF(q^r)$.

Si può dimostrare che un codice lineare a blocco q-ario (n, k) è un sottospazio vettoriale di dimensione k dello spazio vettoriale n-dimensionale costruito su $GF(q)$.

All'interno di ogni campo $GF(q^s)$ esistono q elementi che costituiscono il sottocampo $GF(q)$ con le stesse operazioni definite in $GF(q^s)$. Gli elementi di $GF(q)$ sono gli unici elementi che in $GF(q^s)$ hanno ordine q.

In ogni campo di Galois $GF(q^s)$ esiste un elemento particolare α, detto *elemento primitivo*, avente ordine q^s. Moltiplicando α per se stesso si ottengono tutti gli altri elementi del campo a esclusione dello zero. Esprimendo quindi gli elementi di $GF(q^s)$ come potenze di

α, è possibile riconoscere gli elementi del sottocampo $GF(q)$, poiché questi sono gli elementi 0, 1 e gli altri elementi di $GF(q^s)$ aventi ordine uguale agli elementi di $GF(q)$, cioè i termini con esponente pari a $(q^s-1)/(q-1)$, $2(q^s-1)/(q-1)$, …, $(q-2)(q^s-1)/(q-1)$.

Sia per evidenziare una corrispondenza biunivoca tra gli elementi di un campo $GF(q^s)$ con quelli del sottocampo $GF(q)$, sia per ottenere una descrizione in forma binaria delle operazioni da effettuare durante la codifica e la decodifica, è utile ricorrere alla descrizione polinomiale dei campi di Galois.

Un polinomio $p(x)$ di grado r su $GF(q)$ è un polinomio nell'incognita x i cui coefficienti e il cui termine noto sono elementi di $GF(q)$.

Le operazioni tra due polinomi su $GF(q)$ coincidono con le usuali operazioni tra polinomi, con l'eccezione che le somme e i prodotti tra i coefficienti sono effettuate in $GF(q)$.

Ogni campo esteso $GF(q^s)$ può essere generato a partire dal sottocampo $GF(q)$, ricorrendo alla rappresentazione polinomiale. Esiste infatti almeno un polinomio di grado s su $GF(q)$, grazie al quale è possibile costruire il campo $GF(q^s)$. Tale polinomio è un *polinomio primitivo* di $GF(q)$ (per una trattazione dettagliata dell'argomento si veda il succitato *Theory*

and Practice of Error Control Codes).

Grazie al polinomio primitivo ogni elemento di $GF(q^s)$ può essere rappresentato in forma q-aria mediante un polinomio su $GF(q)$ di grado $(s-1)$, e le operazioni nel campo esteso $GF(q^s)$ possono essere ottenute dalle operazioni di addizione e moltiplicazione tra polinomi su $GF(q)$.

Per convertire una matrice su $GF(q^s)$ in una matrice su $GF(q)$ è sufficiente disporre di una rappresentazione di $GF(q^s)$ su $GF(q)$ e di una rappresentazione di $GF(q^s)$ su $GF(2)$. Posto $q=2^r$, la prima rappresentazione può essere ottenuta mediante un polinomio primitivo di grado s su $GF(q)$, mentre la seconda può essere ottenuta mediante un polinomio primitivo di grado sr su $GF(2)$.

Nell'esempio specifico descritto precedentemente, è stata utilizzata la rappresentazione del campo $GF(16)$ su $GF(4)$ ottenuta con il polinomio quaternario $p_1(x)=x^2+x+2$, la rappresentazione del campo $GF(16)$ su $GF(2)$ ottenuta con il polinomio binario $p_2(x)=x^4+x+1$, entrambe riportate nella Tabella I di figura 1, e la rappresentazione del campo $GF(4)$ su $GF(2)$ ottenuta con il polinomio binario $p_3(x)=x^2+x+1$, riportata nella tabella II di figura 2.

In particolare, nella tabella I, la notazione esadecimale associata alla corrispondente notazione

polinomiale binaria altro non è che la rappresentazione in base esadecimale dei coefficienti dei termini incogniti del rispettivo polinomio binario. In altre parole, per ciascuno dei polinomi binari, si forma una parola binaria con i coefficienti dei vari termini del polinomio binario e si converte tale parola binaria nella base esadecimale, ottenendo così la corrispondente notazione esadecimale.

Analogamente, nella tabella II la notazione quaternaria associata alla corrispondente notazione polinomiale binaria altro non è che la rappresentazione in base quaternaria dei coefficienti dei termini incogniti del rispettivo polinomio binario.

Esaminando la tabella I, si possono riconoscere gli elementi di *GF*(16) che sono elementi di *GF*(4). In notazione esponenziale essi corrispondono agli elementi 0, 1, α^5 e α^{10}, mentre in notazione esadecimale (ottenuta dalla notazione polinomiale binaria convertendo in base esadecimale i coefficienti dei polinomi associati a ciascun polinomio), sono gli elementi 0, 1, 6 e 7.

In particolare, si nota come, a parte 0 e 1, la cui associazione è evidente ed immediata, sia α^5 e α^{10} hanno molteplicità quattro e pertanto, le sole rappresentazioni polinomiali binarie di *GF*(16) su *GF*(2) e di *GF*(4) su *GF*(2) non sono sufficienti per stabilire

se la corrispondenza esatta sia 2→6 e 3→7 oppure sia 2→7 e 3→6. Attraverso però la rappresentazione polinomiale di $GF(16)$ su $GF(4)$, l'associazione corretta risulta automatica, come mostrato nella tabella I.

Si può dimostrare che, fissati n, k e t, se il limite di Hamming definito dalla relazione (1) è soddisfatto per $q=p^r$, allora lo stesso limite è soddisfatto anche per $q=p^{rs}$ (con p primo, r e s interi positivi).

Pertanto, è sempre possibile convertire la matrice generatrice di un codice lineare a blocco (n, k) su $GF(p^r)$ nella matrice generatrice di un codice lineare a blocco (n, k) su un qualsiasi campo esteso $GF(p^{rs})$.

L'estensione può essere effettuata mappando ogni elemento di $GF(p^r)$, che compare nella matrice di partenza, nel corrispondente elemento del campo esteso. La corrispondenza tra gli elementi dei due campi è espressa nella rappresentazione di $GF(p^{rs})$ su $GF(p^r)$, a cui va affiancata una rappresentazione di $GF(p^{rs})$ su $GF(p)$, che consente la realizzazione in logica p-aria delle operazioni di codifica e decodifica.

In particolare, per ogni codice lineare a blocco su $GF(2^{rs})$ le operazioni di codifica e decodifica possono essere effettuate in logica binaria ricorrendo alla rappresentazione polinomiale binaria degli elementi del

campo.

La matrice generatrice ottenuta genera un codice (n, k) su $GF(p^{rs})$ che risulta *isomorfo* a un codice (n, k) su $GF(p^r)$ avente almeno la stessa capacità di correzione del codice sul campo esteso. Si possono così sfruttare gli stessi circuiti di codifica e di decodifica per parole con lo stesso numero di simboli su $GF(p^r)$ o su $GF(p^{rs})$.

Per realizzare i circuiti di codifica e decodifica in logica binaria è necessario disporre di una descrizione delle operazioni di $GF(p^{rs})$ in termini di operazioni tra bit. Ricorrendo alla rappresentazione in forma polinomiale binaria, in ogni *campo di Galois* con un numero di elementi pari a una potenza di 2 si possono effettuare tutte le operazioni di addizione e moltiplicazione tra gli elementi del campo mediante le operazioni di addizione e moltiplicazione in $GF(2)$. Tali operazioni coincidono rispettivamente con le operazioni logiche di *exor* (or esclusivo) e di *and*.

Ogni operazione di somma tra due elementi di $GF(2^m)$ può essere quindi realizzata semplicemente effettuando in $GF(2)$ le $m+1$ somme tra i coefficienti dei termini di pari grado dei due polinomi.

La moltiplicazione di un elemento incognito $w(x)$ di $GF(2^m)$ per un determinato elemento $e(x)$ dello stesso

campo può essere realizzata in forma binaria ricorrendo alla rappresentazione polinomiale nel modo seguente:

$$y(x)=R_{p(x)}\{w(x)\,e(x)\}=y_{m-1}x^{m-1}+y_{m-2}x^{m-2}+\ldots+y_0$$

dove i coefficienti y_i sono coefficienti binari, $p(x)$ è il polinomio primitivo binario con cui è costruito $GF(2^m)$ e $R_{p(x)}\{g(x)\}$ indica il resto della divisione del polinomio $g(x)$ per il polinomio $p(x)$.

A questo punto si può notare che la realizzazione circuitale dei blocchi, che effettuano le moltiplicazioni per gli elementi della matrice generatrice diversi da 0 e 1, cambia al variare del polinomio primitivo binario utilizzato per rappresentare il campo esteso. Ad esempio, tornando all'esempio esaminato precedentemente, la rappresentazione polinomiale binaria del campo $GF(16)$ ottenuta mediante il polinomio $p_4(x)=x^4+x^3+1$, riportata nella tabella III di figura 3, richiede un diverso numero di porte logiche sia per effettuare la moltiplicazione per α^5, sia per effettuare la moltiplicazione per α^{10}.

Infatti, per effettuare la moltiplicazione di un elemento incognito γ per α^5, nel caso in cui si scelga la rappresentazione polinomiale riportata nella tabella I, si procede nel modo seguente:

$$\gamma=\gamma_3 x^3+\gamma_2 x^2+\gamma_1 x+\gamma_0$$

$$\gamma\cdot\alpha^5=R_{p_2(x)}\{\gamma_3 x^5+(\gamma_3+\gamma_2)\,x^4+(\gamma_2+\gamma_1)\,x^3+(\gamma_1+\gamma_0)\,x^2+\gamma_0 x\}$$

dove $R_{p_2(x)}\{g(x)\}$ indica il resto della divisione del polinomio $g(x)$ per il polinomio $p_2(x)$. Si ottiene quindi:

$$\gamma\cdot\alpha^5=(\gamma_2+\gamma_1)\,x^3+(\gamma_3+\gamma_1+\gamma_0)\,x^2+(\gamma_2+\gamma_0)\,x+(\gamma_3+\gamma_2)$$

Le operazioni di somma tra i coefficienti γ_i sono addizioni binarie e quindi possono essere realizzate con porte logiche di tipo exor.

Analogamente, per la moltiplicazione per α^{10} si ha:

$$\gamma\cdot\alpha^{10}=(\gamma_3+\gamma_2+\gamma_1)\,x^3+(\gamma_3+\gamma_2+\gamma_1+\gamma_0)\,x^2+(\gamma_2+\gamma_1+\gamma_0)\,x+(\gamma_3+\gamma_2+\gamma_0)$$

Pertanto, le moltiplicazioni per α^5 ed α^{10} in $GF(16)$ possono essere effettuate con i circuiti mostrati rispettivamente nelle figure 4a e 4b.

Sfruttando invece la rappresentazione di tabella III, la moltiplicazione di γ per α^5 è descritta dal polinomio:

$$\gamma\cdot\alpha^5=(\gamma_3+\gamma_1+\gamma_0)\,x^3+(\gamma_3+\gamma_2+\gamma_1)\,x^2+(\gamma_3+\gamma_2+\gamma_1+\gamma_0)\,x+(\gamma_2+\gamma_1+\gamma_0)$$

e la moltiplicazione di γ per α^{10} è descritta da:

$$\gamma\cdot\alpha^{10}=(\gamma_1+\gamma_0)\,x^3+(\gamma_3+\gamma_1)\,x^2+(\gamma_3+\gamma_2+\gamma_0)\,x+(\gamma_2+\gamma_1)$$

I relativi circuiti di moltiplicazioni sono

mostrati rispettivamente nelle figure 5a e 5b.

Si noti che non sempre occorre utilizzare un exor per ogni segno "+" presente tra i coefficienti nell'equazione che fornisce il prodotto. Infatti le somme, anche se in posizioni diverse, possono coinvolgere gli stessi coefficienti ed è quindi possibile ridurre il numero di porte logiche.

Confrontando le figure 4a, 4b e 5a, 5b appare evidente che, per quanto riguarda la moltiplicazione per l'elemento α^5, la prima rappresentazione richiede un numero minore di porte exor (cinque contro sei).

Per ottenere un codice a controllo dell'errore su $GF(q^s)$ con determinate specifiche in termini di k e t, che possa essere utilizzato anche su $GF(q)$, va dapprima calcolata la matrice generatrice (o la matrice del controllo di parità) di un codice su $GF(q)$. Per ottenere la matrice si può fare ricorso a una famiglia nota di codici, quali ad esempio i codici di Hamming e i codici BCH (per una trattazione dettagliata dell'argomento si veda ad esempio S. Benedetto, E. Biglieri, V. Castellani, *Digital Trasmission Theory*, Prentice Hall, Englewood Cliffs (New Jersey), 1987).

Successivamente si abbrevia la matrice ottenuta al valore di k voluto. Per l'abbreviazione della matrice si devono prendere in considerazione le possibili

rappresentazioni di $GF(q^s)$ su $GF(2)$ e valutare quale di queste permette la realizzazione dei blocchi circuitali di area minore. Una volta scelta la rappresentazione si può procedere all'abbreviazione del codice eliminando le colonne della matrice del controllo di parità contenenti il maggior numero di simboli di $GF(q^s)$ per i quali la moltiplicazione risulta più dispendiosa in termini di area.

Tornando all'esempio considerato precedentemente, la matrice generatrice $\mathbf{G}_1$ è stata ottenuta abbreviando la matrice generatrice del codice di Hamming (21, 18) su $GF(4)$. Per minimizzare il numero di porte logiche che compongono il circuito di codifica, l'abbreviazione deve procedere con l'intento di minimizzare il numero di simboli 2 e 3 (ovvero α^5 e α^{10}). Dopo aver scelto il polinomio binario per la rappresentazione binaria di $GF(16)$, si può determinare quale simbolo tra α^5 e α^{10} richiede il minor numero di porte logiche. Tale simbolo va preferito durante l'abbreviazione.

Nell'esempio preso in considerazione, scegliendo come polinomio primitivo per la costruzione di $GF(16)$ il polinomio $p_2(x)$, vanno preferiti i termini in α^5 che richiedono 5 porte exor contro le 6 porte exor richieste dalla moltiplicazione per α^{10}. La matrice generatrice più conveniente in termini di risparmio di area risulta

essere la matrice G_2 sopra scritta. Al contrario, sfruttando la rappresentazione di $GF(16)$ ottenuta mediante il polinomio $p_3(x)$ risulta più vantaggioso utilizzare la matrice G_3 di seguito indicata, poiché in questo caso la moltiplicazione per α^{10}=A richiede cinque porte exor contro le sei porte exor di α^5=B.

$$G_3 = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & B \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & A \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & B \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & A \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & B \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & A \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & B & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & B & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & A & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & A & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & A & B \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & A & A
\end{pmatrix}$$

In generale, quindi, dati i valori di k e t, la procedura da seguire per costruire un codice a controllo dell'errore in grado di operare su celle di memoria funzionanti a un numero di livelli che può assumere tutti i valori dell'insieme $\{b^{a_1}, b^{a_1 a_2}, \ldots, b^{a_1 a_2 \cdots a_h}\}$, con b, $a_1, \ldots, a_h$ interi positivi, è la seguente (si faccia riferimento anche al diagramma di flusso mostrato nella figura 6).

1. Acquisiti k, t, b^{a_1}, $b^{a_1 a_2}$, …, $b^{a_1 a_2 \cdots a_h}$, che rappresentano le specifiche di progetto del codice a

controllo dell'errore che si vuole ottenere (blocco 10), si calcola il minimo valore di n tale che sia soddisfatto il limite di Hamming definito dalla relazione (1) sopra riportata (blocco 20).

2. Dati n e k, si calcolano i valori massimi di n e k, indicati nel seguito rispettivamente con $\hat{n}$ e $\hat{k}$, che soddisfano il limite di Hamming per $q=b^{a_1}$, t e $(\hat{n}-\hat{k})=(n-k)$ (blocco 30).

In altre parole, dato che il valore di n calcolato al punto precedente con la relazione (1) rappresenta il numero di simboli di una parola di codice di un codice lineare q-ario abbreviato e che, quindi, gli $n-k$ simboli di parità di tale codice lineare abbreviato rappresentano il minimo numero di simboli di controllo richiesti affinché il codice abbia capacità di correzione t, nel punto 2, si utilizza nuovamente la relazione (1) per calcolare, dati i valori di n e k del codice lineare q-ario abbreviato, i valori $\hat{n}$ e $\hat{k}$ del corrispondente codice lineare q-ario non abbreviato.

3. Si determina la matrice generatrice del codice abbreviato $(\hat{n}, \hat{k})$ su $GF(b^{a_1})$ con t dato (blocco 40).

4. Si costruiscono le rappresentazioni esponenziali dei campi $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_h})$ (blocco 50).

5. Sulla base delle rappresentazioni esponenziali di cui al punto precedente, si identificano gli elementi

del campo $GF(b^{a_1 a_2 \cdots a_n})$ che fanno parte anche dei campi $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_{n-1}})$ (blocco 60).

In particolare, gli elementi di $GF(b^{a_1 a_2 \cdots a_n})$ che fanno parte anche di $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_{n-1}})$ sono quelli che hanno una molteplicità rispettivamente pari a b^{a_1}, $b^{a_1 a_2}$, ..., $b^{a_1 a_2 \cdots a_{n-1}}$. Nell'esempio specifico preso in considerazione in precedenza, gli elementi di $GF(16)$ che fanno parte di $GF(4)$ sono i simboli 0, 1, α^5 e α^{10}, in quanto sono gli unici che hanno una molteplicità pari a quattro, ossia sono gli unici che, moltiplicati quattro volte per se stessi, forniscono come risultato sempre se stessi.

Si sottolinea inoltre il fatto che in letteratura la definizione di molteplicità non è univoca. In particolare, in alcuni testi è definita, analogamente a quanto fatto in questa trattazione, come il numero di volte che un elemento del campo a elementi finiti deve essere moltiplicato per se stesso per ottenere nuovamente tale elemento, mentre in altri testi la molteplicità è una funzione lineare di tale numero. Pertanto, l'identificazione degli elementi di $GF(b^{a_1 a_2 \cdots a_n})$ che fanno parte anche di $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_{n-1}})$ può anche essere fatta utilizzando una definizione di molteplicità differente da quella utilizzata in questa trattazione, senza per questo modificare in alcun modo

quanto descritto.

6. Si stabiliscono quindi le corrispondenze biunivoche fra gli elementi del campo $GF(b^{a_1 a_2 \cdots a_h})$ e gli elementi dei campi $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_{h-1}})$ ad essi isomorfi identificati nel punto precedente (blocco 70), e si sostituisce quindi ciascuno degli elementi della matrice generatrice su $GF(b^{a_1})$ di cui al punto 3 con il corrispondente elemento isomorfo appartenente a $GF(b^{a_1 a_2 \cdots a_h})$, ottenendo così una matrice generatrice polivalente che, assieme alle succitate corrispondenze biunivoche fra gli elementi dei campi $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_{h-1}})$ e gli elementi del campo $GF(b^{a_1 a_2 \cdots a_h})$ ad essi isomorfi, definisce il codice a controllo dell'errore polivalente desiderato (blocco 80).

In particolare, per stabilire le corrispondenze biunivoche fra gli elementi dei campi $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_{h-1}})$ e gli elementi del campo $GF(b^{a_1 a_2 \cdots a_h})$ ad essi isomorfi, si costruiscono innanzitutto le rappresentazioni polinomiali binarie di tutti i campi $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_h})$ mediante rispettivi polinomi primitivi di grado rispettivamente pari a_1, $a_1 a_2$, ..., $a_1 a_2 \ldots a_h$ su $GF(2)$, attraverso le quali a ogni elemento di $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_h})$ viene associato un corrispondente polinomio binario di grado minore della rispettiva base, ossia a ogni elemento di

$GF(b^{a_1})$ viene associato un corrispondente polinomio binario di grado minore di a_1, a ogni elemento di $GF(b^{a_1 a_2})$ viene associato un corrispondente polinomio binario di grado minore di $a_1 a_2$, e così via fino a $GF(b^{a_1 a_2 \cdots a_h})$, a ogni elemento del quale viene associato un corrispondente polinomio binario di grado minore di $a_1 a_2 \ldots a_h$.

Le corrispondenze biunivoche fra gli elementi del campo $GF(b^{a_1 a_2 \cdots a_h})$ e gli elementi dei campi $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_{h-1}})$ e ad essi isomorfi vengono quindi stabilite utilizzando tali rappresentazioni polinomiali binarie, nel modo seguente.

Per ogni elemento di $GF(b^{a_1 a_2 \cdots a_h})$ isomorfo a un elemento di $GF(b^{a_1})$, si forma una prima parola binaria con i coefficienti dei termini incogniti del polinomio binario associato all'elemento di $GF(b^{a_1 a_2 \cdots a_h})$ e si converte tale parola binaria nella base $b^{a_1 a_2 \cdots a_h}$, ottenendo un primo simbolo; si forma quindi una seconda parola binaria con i coefficienti dei termini incogniti del polinomio binario associato all'elemento di $GF(b^{a_1})$ e si converte tale parola binaria in una cifra nella base b^{a_1}, ottenendo così un secondo simbolo; i due simboli così ottenuti definiscono la corrispondenza biunivoca cercata fra l'elemento di $GF(b^{a_1 a_2 \cdots a_h})$ e l'elemento di $GF(b^{a_1})$ a esso isomorfo.

Ripetendo quanto sopra descritto per tutti gli elementi di $GF(b^{a_1 a_2 \cdots a_h})$ isomorfi agli elementi di $GF(b^{a_1})$, per tutti gli elementi di $GF(b^{a_1 a_2 \cdots a_h})$ isomorfi agli elementi di $GF(b^{a_1 a_2})$, e così via fino a tutti gli elementi di $GF(b^{a_1 a_2 \cdots a_h})$ isomorfi agli elementi di $GF(b^{a_1 a_2 \cdots a_{h-1}})$, si ottengono le corrispondenze biunivoche cercate fra gli elementi del campo $GF(b^{a_1 a_2 \cdots a_h})$ e gli elementi dei campi $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, …, $GF(b^{a_1 a_2 \cdots a_{h-1}})$ e ad essi isomorfi.

Nell'esempio specifico preso in considerazione in precedenza (si veda la tabella I), il simbolo corrispondente ad ognuno dei quattro elementi 0, 1, α^5 e α^{10} altro non è che la rappresentazione in base esadecimale della parola binaria formata dai coefficienti dei termini incogniti del polinomio binario associato a tale elemento.

Qualora vi siano più elementi del sottocampo ad avere la medesima molteplicità, l'associazione fra tali elementi e quelli isomorfi del sottocampo corrispondente può essere effettuata ricorrendo alla rappresentazione polinomiale del campo $GF(b^{a_1 a_2 \cdots a_h})$ effettuata con polinomi costruiti sul sottocampo.

Ad esempio, se più elementi del sottocampo $GF(b^{a_1 a_2 \cdots a_i})$, con $i \leq h$, hanno la stessa molteplicità, allora l'associazione fra tali elementi di $GF(b^{a_1 a_2 \cdots a_i})$ e quelli ad essi isomorfi di $GF(b^{a_1 a_2 \cdots a_h})$ può essere effettuata

costruendo la rappresentazione polinomiale di $GF(b^{a_1 a_2 \cdots a_h})$ su $GF(b^{a_1 a_2 \cdots a_i})$, ossia ricavata mediante un polinomio primitivo di grado $a_{i+1} \cdots a_h$ su $GF(b^{a_1 a_2 \cdots a_i})$. Tale rappresentazione polinomiale fornisce automaticamente l'associazione cercata.

Nell'esempio specifico preso precedentemente in considerazione in questa trattazione (si veda ancora la tabella I), vi sono due elementi di $GF(4)$ ad avere molteplicità pari a quattro: questi elementi sono α^5 e α^{10}. Pertanto, le sole rappresentazioni polinomiali binarie di $GF(16)$ e di $GF(4)$ non sono sufficienti per stabilire se la corrispondenza esatta sia 2→6 e 3→7 oppure sia 2→7 e 3→6. Attraverso però la rappresentazione polinomiale di $GF(16)$ su $GF(4)$, l'associazione corretta risulta automatica, come mostrato nella tabella I.

Per quanto riguarda invece la sostituzione degli elementi della matrice generatrice di $GF(b^{a_1})$ con il corrispondente elemento isomorfo appartenente a $GF(b^{a_1 a_2 \cdots a_h})$, questa viene effettuata utilizzando i simboli, in base b^{a_1} e $b^{a_1 a_2 \cdots a_h}$, rappresentativi degli elementi di $GF(b^{a_1 a_2 \cdots a_h})$ isomorfi agli elementi di $GF(b^{a_1})$.

In particolare, ciascuno dei simboli nella base b^{a_1} presenti nella matrice generatrice su $GF(b^{a_1})$ viene sostituito con il corrispondente simbolo nella base

$b^{a_1 a_2 \cdots a_n}$, ottenendo così una matrice generatrice polivalente che definisce un codice a controllo dell'errore utilizzabile con celle di memoria i cui livelli di memorizzazione possono assumere i valori dell'insieme $\{b^{a_1}, b^{a_1 a_2}, \ldots, b^{a_1 a_2 \cdots a_n}\}$.

7. A questo punto, si determina, per ogni elemento della matrice generatrice polivalente (rappresentativo di un corrispondente elemento di $GF(b^{a_1 a_2 \cdots a_n})$), il numero di porte logiche richiesto per effettuare l'operazione di moltiplicazione associata a tale elemento, ricorrendo alla rappresentazione polinomiale binaria (blocco 90).

8. Si abbrevia la matrice generatrice polivalente ottenuta al valore di k inizialmente specificato, minimizzando il numero di elementi di $GF(b^{a_1 a_2 \cdots a_n})$ che richiedono il maggior numero di porte logiche (blocco 100).

In altre parole, essendo la matrice generatrice **G=[I, P]** con $\hat{k}$ righe e $\hat{n}$ colonne, dove **I** è la matrice identità di dimensione $\hat{k}$ mentre **P** è una matrice $\hat{k} \times (\hat{n} - \hat{k})$, si eliminano le righe della matrice **P** contenenti elementi che richiedono il maggior numero di porte logiche, fino ad ottenere una matrice generatrice abbreviata avente un numero di righe pari al valore di k specificato nel punto 1.

9. Dalla matrice generatrice polivalente abbreviata

così ottenuta si possono generare i circuiti di codifica e decodifica nel modo consueto descritto ad esempio nei succitati *Error-Correcting Codes* e *Theory and Practice of Error Control Codes*. Tale matrice genera sempre un codice su $GF(b^{a_1 a_2 \cdots a_h})$: quando le celle funzionano con b^{a_1} livelli il codice a controllo dell'errore generato dalla matrice generatrice polivalente abbreviata è un codice a controllo dell'errore su $GF(b^{a_1 a_2 \cdots a_h})$ isomorfo a un codice a controllo dell'errore su $GF(b^{a_1})$ che soddisfa i requisiti iniziali, mentre quando le celle funzionano con $b^{a_1 \cdots a_i}$ livelli e $1<i<h$, il codice a controllo dell'errore generato dalla matrice generatrice polivalente abbreviata è un codice a controllo dell'errore su $GF(b^{a_1 a_2 \cdots a_h})$ isomorfo a un codice a controllo dell'errore su $GF(b^{a_1 \cdots a_i})$ che soddisfa i requisiti iniziali, e così via (blocco 110).

Nelle figure 7 e 8 sono mostrati due schemi a blocchi rappresentativi delle operazioni di codifica, memorizzazione e decodifica di una parola di informazione effettuate utilizzando un codice a controllo dell'errore non polivalente secondo l'arte nota e, rispettivamente, un codice a controllo dell'errore polivalente secondo la presente invenzione.

Come si può notare analizzando la figura 7, l'implementazione di un metodo di controllo dell'errore

non polivalente secondo l'arte nota operante su celle di memoria funzionanti solo a 2^r livelli di memorizzazione (memorizzanti cioè r bit) prevede l'uso di un codificatore 120 che aggiunge ai k simboli 2^r-ari delle parole di informazione di ingresso $(n-k)$ simboli 2^r-ari per il controllo dell'errore. Le parole di codice così ottenute vengono scritte nella matrice di memoria 122, e da questa lette, mediante circuiti di scrittura e di lettura 124, 126 noti che effettuano la scrittura e la lettura di n celle di memoria a 2^r livelli di memorizzazione in un ciclo.

Le parole lette vengono poi decodificate da un decodificatore 128 che, sulla base degli n simboli 2^r-ari della parola letta, effettua il controllo dell'errore e restituisce in uscita parole di k simboli 2^r-ari che sono la stima delle parole di informazione originarie.

Come si può notare invece confrontando la figura 8 con la figura 7, l'implementazione di un metodo di controllo dell'errore polivalente secondo la presente invenzione operante su celle di memoria funzionanti a 2^{sr} livelli di memorizzazione (ad esempio, con s=1 o s=2 ed r fisso, celle di memoria memorizzanti r bit oppure $2r$ bit rispettivamente), prevede l'uso di un transcodificatore d'ingresso 130 che riceve in ingresso

un segnale di controllo S indicativo del numero di livelli di memorizzazione a cui funzionano le celle di memoria e trasforma quindi, simbolo per simbolo, gruppi di sr bit in gruppi di $2r$ bit sulla base delle corrispondenze biunivoche determinate, ossia converte, simbolo per simbolo, le parole di informazione dalla base in cui sono rappresentate, che è pari al numero di livelli di memorizzazione a cui funzionano le celle di memoria, a una base pari al numero massimo di livelli di memorizzazione delle celle di memoria, utilizzando le suddette corrispondenze biunivoche

Un codificatore 132 aggiunge quindi ai k simboli 2^{2r}-ari delle parole di informazione di ingresso $(n-k)$ simboli 2^{2r}-ari per il controllo dell'errore. Le parole di codice così ottenute vengono scritte nella matrice di memoria 134, e da questa lette, mediante circuiti di scrittura e di lettura 136, 138 noti che ricevono in ingresso il suddetto segnale di controllo S ed effettuano la scrittura e la lettura di n celle di memoria a 2^{sr} livelli di memorizzazione in un ciclo.

Le parole lette vengono poi decodificate mediante un decodificatore 140 che, sulla base degli n simboli 2^{2r}-ari della parola letta, effettua il controllo dell'errore e restituisce in uscita parole di k simboli 2^{2r}-ari che sono la stima delle parole di informazione

originarie.

Un transcodificatore di uscita 142 ricevente anch'esso in ingresso il suddetto segnale di controllo S trasforma infine, simbolo per simbolo, gruppi di $2r$ bit in gruppi di sr bit, effettuando la conversione inversa rispetto al transcodificatore di ingresso 120, ossia convertendo, simbolo per simbolo, le parole decodificate, da una base pari al numero massimo di livelli di memorizzazione di dette celle di memoria a una base pari al numero di livelli di memorizzazione a cui funzionano le celle di memoria, utilizzando le suddette corrispondenze biunivoche.

Da un esame delle caratteristiche del metodo polivalente di correzione dell'errore realizzato secondo la presente invenzione sono evidenti i vantaggi che esso consente di ottenere.

In particolare, il codice a controllo dell'errore costruito nel modo sopra descritto presenta la capacità di controllare l'errore in celle di memoria multilivello funzionanti a un numero variabile di livelli di memorizzazione utilizzando gli stessi blocchi circuitali.

Inoltre, il codice a controllo dell'errore costruito nel modo sopra descritto presenta il massimo livello di affidabilità in ogni modalità di

funzionamento, consente un migliore sfruttamento delle celle di memoria utilizzate per il controllo, richiede un aggravio di area su silicio minimo e introduce tempi di ritardo aggiuntivi praticamente nulli.

Risulta infine chiaro che al metodo di costruzione ed al metodo polivalente di controllo dell'errore qui descritti ed illustrati possono essere apportate modifiche e varianti senza per questo uscire dall'ambito protettivo della presente invenzione, come definito nelle rivendicazioni allegate.

R I V E N D I C A Z I O N I

1. Metodo di costruzione di un codice a controllo dell'errore polivalente per celle di memoria multilivello funzionanti a un numero variabile di livelli di memorizzazione, in particolare celle di memoria i cui livelli di memorizzazione possono assumere i valori dell'insieme $\{b^{a_1}, b^{a_1 a_2}, \ldots, b^{a_1 a_2 \cdots a_n}\}$, detto codice a controllo dell'errore codificando parole di informazione (**i**), formate da k simboli q-ari appartenenti a un alfabeto contenente q simboli distinti, con $q \in \{b^{a_1}, b^{a_1 a_2}, \ldots, b^{a_1 a_2 \cdots a_n}\}$, in corrispondenti parole di codice (**c**) formate da n simboli q-ari, con $q = b^{a_1 a_2 \cdots a_n}$, ed avente una capacità di correzione dell'errore t, ogni parola di codice (**c**) essendo generata attraverso una operazione di moltiplicazione fra la corrispondente parola di informazione (**i**) ed una matrice generatrice (**G**); detto metodo di costruzione comprendendo le fasi di:

- acquisire i valori di k, t, b^{a_1}, $b^{a_1 a_2}$, ..., $b^{a_1 a_2 \cdots a_n}$, che costituiscono le specifiche di progetto di detto codice a controllo dell'errore;

- calcolare, in funzione di $q = b^{a_1}$, k e t, il minimo valore di n tale che sia soddisfatto il limite di Hamming;

- calcolare i valori massimi $\hat{n}$ e $\hat{k}$ di n e k che

soddisfano il detto limite di Hamming per $q=b^{a_1}$, t e $(\hat{n}-\hat{k})=(n-k)$;

- determinare, in funzione di t, la matrice generatrice del codice a controllo dell'errore $(\hat{n}, \hat{k})$ sul campo a elementi finiti $GF(b^{a_1})$;

- costruire rappresentazioni polinomiali binarie dei campi a elementi finiti $GF(b^{a_1})$, $GF(b^{a_1a_2})$, ..., $GF(b^{a_1a_2\cdots a_h})$;

- identificare, utilizzando dette rappresentazioni esponenziali, gli elementi del campo a elementi finiti $GF(b^{a_1a_2\cdots a_h})$ isomorfi agli elementi dei campi a elementi finiti $GF(b^{a_1})$, $GF(b^{a_1a_2})$, ..., $GF(b^{a_1a_2\cdots a_{h-1}})$;

- stabilire corrispondenze biunivoche fra gli elementi dei campi a elementi finiti $GF(b^{a_1})$, $GF(b^{a_1a_2})$, ..., $GF(b^{a_1a_2\cdots a_{h-1}})$ e gli elementi del campo a elementi finiti $GF(b^{a_1a_2\cdots a_h})$ ad essi isomorfi; e

- sostituire ciascuno degli elementi della detta matrice generatrice con il corrispondente elemento isomorfo del campo a elementi finiti $GF(b^{a_1a_2\cdots a_h})$, ottenendo così una matrice generatrice polivalente definente, assieme a dette corrispondenze biunivoche, detto codice a controllo dell'errore polivalente utilizzabile con celle di memoria i cui livelli di memorizzazione possono assumere i valori dell'insieme $\{b^{a_1}, b^{a_1a_2}, \ldots, b^{a_1a_2\cdots a_h}\}$.

2. Metodo di costruzione secondo la rivendicazione 1, in cui detto codice a controllo dell'errore è un codice lineare a blocco.

3. Metodo di costruzione secondo la rivendicazione 1 o 2, in cui l'identificazione degli elementi del campo a elementi finiti $GF(b^{a_1 a_2 \cdots a_n})$ isomorfi agli elementi dei campi a elementi finiti $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_{n-1}})$ viene effettuata sulla base della molteplicità di detti elementi nel campo a elementi finiti $GF(b^{a_1 a_2 \cdots a_n})$.

4. Metodo di costruzione secondo la rivendicazione 3, in cui detta fase di identificare gli elementi del campo a elementi finiti $GF(b^{a_1 a_2 \cdots a_n})$ isomorfi agli elementi dei campi a elementi finiti $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_{n-1}})$ comprende la fase di individuare gli elementi del campo a elementi finiti $GF(b^{a_1 a_2 \cdots a_n})$ che hanno rispettivamente molteplicità pari a, o funzione di, b^{a_1}, $b^{a_1 a_2}$, ..., $b^{a_1 a_2 \cdots a_{n-1}}$.

5. Metodo di costruzione secondo una qualsiasi delle rivendicazioni precedenti, in cui detta fase di stabilire corrispondenze biunivoche fra gli elementi dei campi a elementi finiti $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_{n-1}})$ e gli elementi del campo a elementi finiti $GF(b^{a_1 a_2 \cdots a_n})$ ad essi isomorfi comprende le fasi di:

- costruire rappresentazioni polinomiali binarie dei campi a elementi finiti $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ...,

$GF(b^{a_1 a_2 \cdots a_h})$; e

- stabilire corrispondenze biunivoche fra gli elementi dei campi a elementi finiti $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_{h-1}})$ e gli elementi del campo a elementi finiti $GF(b^{a_1 a_2 \cdots a_h})$ ad essi isomorfi, utilizzando dette rappresentazioni polinomiali binarie.

6. Metodo di costruzione secondo la rivendicazione 5, in cui dette rappresentazioni polinomiali binarie dei campi a elementi finiti $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_h})$ sono costruite utilizzando rispettivi polinomi primitivi di grado rispettivamente pari a a_1, $a_1 a_2$, ..., $a_1 a_2 \ldots a_h$ sul campo a elementi finiti $GF(2)$, dette rappresentazioni polinomiali binarie associando a ogni elemento dei rispettivi campi a elementi finiti $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_h})$ un corrispondente polinomio binario di grado rispettivamente minore di a_1, $a_1 a_2$, ..., $a_1 a_2 \ldots a_h$.

7. Metodo di costruzione secondo la rivendicazione 6, in cui detta fase di stabilire corrispondenze biunivoche fra gli elementi del campo a elementi finiti $GF(b^{a_1 a_2 \cdots a_h})$ e gli elementi dei campi a elementi finiti $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_{h-1}})$ ad essi isomorfi comprende le fasi di:

- per ogni elemento del campo a elementi finiti $GF(b^{a_1 a_2 \cdots a_h})$ isomorfo a un corrispondente un elemento di uno dei campi a elementi finiti $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ...,

$GF(b^{a_1 a_2 \cdots a_{n-1}})$, formare una prima parola binaria con i coefficienti dei termini del polinomio binario associato a detto elemento di $GF(b^{a_1 a_2 \cdots a_n})$, ed una seconda parola binaria con i coefficienti dei termini del polinomio binario associato al detto corrispondente elemento di uno dei campi a elementi finiti $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_{n-1}})$;

- convertire detta prima parola binaria nella base $b^{a_1 a_2 \cdots a_n}$, ottenendo così un primo simbolo, e detta seconda parola binaria nella base del campo a elementi finiti a cui appartiene detto corrispondente elemento, ottenendo così un secondo simbolo; detti primo e secondo simbolo definendo la corrispondenza biunivoca fra detto elemento del campo a elementi finiti $GF(b^{a_1 a_2 \cdots a_n})$ ed il corrispondente elemento di uno dei campi a elementi finiti $GF(b^{a_1})$, $GF(b^{a_1 a_2})$, ..., $GF(b^{a_1 a_2 \cdots a_{n-1}})$ a esso isomorfo.

8. Metodo di costruzione secondo una qualsiasi delle rivendicazioni precedenti, comprendente inoltre le fasi di:

- determinare, per ogni elemento della matrice generatrice polivalente, il numero di porte logiche necessarie per effettuare l'operazione di moltiplicazione associata a tale elemento; e

- abbreviare detta matrice generatrice polivalente al valore di k inizialmente specificato, minimizzando

così il numero di elementi del campo a elementi finiti $GF(b^{a_1 a_2 \cdots a_n})$ che richiedono il maggior numero di porte logiche.

9. Metodo di costruzione secondo la rivendicazione 8, in cui detta fase di determinare il numero di porte logiche viene effettuata utilizzando detta rappresentazione polinomiale binaria del campo a elementi finiti $GF(b^{a_1 a_2 \cdots a_n})$.

10. Metodo polivalente di controllo dell'errore per celle di memoria multilivello funzionanti a un numero variabile di livelli di memorizzazione, in particolare celle di memoria i cui livelli di memorizzazione possono assumere i valori dell'insieme $\{b^{a_1}, b^{a_1 a_2}, \ldots, b^{a_1 a_2 \cdots a_n}\}$, detto metodo comprendendo le fasi di:

- utilizzare un codice a controllo dell'errore secondo una qualsiasi delle rivendicazioni precedenti, detto codice a controllo dell'errore codificando parole di informazione (**i**), formate da k simboli q-ari appartenenti a un alfabeto contenente q simboli distinti, con $q \in \{b^{a_1}, b^{a_1 a_2}, \ldots, b^{a_1 a_2 \cdots a_n}\}$, in corrispondenti parole di codice (**c**) formate da n simboli q-ari, con $q = b^{a_1 a_2 \cdots a_n}$, ed avente una capacità di correzione dell'errore t, ogni parola di codice (**c**) essendo generata attraverso una operazione di moltiplicazione fra la corrispondente parola di informazione (**i**) ed una matrice generatrice

(G);

- convertire, simbolo per simbolo, dette parole di informazione dalla base in cui sono rappresentate, che è pari al numero di livelli di memorizzazione a cui funzionano dette celle di memoria, a una base pari al numero massimo di livelli di memorizzazione di dette celle di memoria, utilizzando dette corrispondenze biunivoche;

- codificare dette parole convertite utilizzando detto codice a controllo dell'errore per ottenere rispettive parole di codice;

- memorizzare dette parole di codice in una memoria;

- decodificare parole lette in detta memoria utilizzando detto codice a controllo dell'errore; e

- convertire, simbolo per simbolo, dette parole decodificate da una base pari al numero massimo di livelli di memorizzazione di dette celle di memoria a una base pari al numero di livelli di memorizzazione a cui funzionano dette celle di memoria, utilizzando dette corrispondenze biunivoche.

11. Metodo di costruzione di un codice a controllo dell'errore polivalente per celle di memoria multilivello funzionanti a un numero variabile di livelli di memorizzazione, sostanzialmente come

descritto con riferimento ai disegni allegati.

12. Metodo polivalente di controllo dell'errore per celle di memoria multilivello funzionanti a un numero variabile di livelli di memorizzazione

p.i.: STMICROELECTRONICS S.R.L.

BERGADANO MIRKO
(Iscritto all'Albo n. 843B)

C.C.I.A.A. Torino

BERGADANO MIRKO
(Iscritto all'Albo n. 843B)

TO 2000A 001049

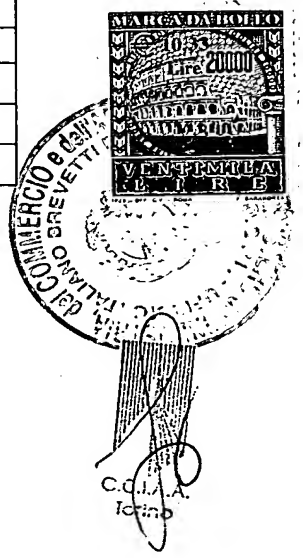
Notazione Esponenziale	Notazione polinomiale quaternaria	Notazione polinomiale binaria di $GF(16)$ su $GF(2)$	Notazione Esadecimale
0	0	0	0
α^0	1	1	1
α^1	x	x	2
α^2	$x+2$	x^2	4
α^3	$3x+2$	x^3	8
α^4	$x+1$	$x+1$	3
α^5	2	x^2+x	6
α^6	$2x$	x^3+x^2	C
α^7	$2x+3$	x^3+x+1	B
α^8	$x+3$	x^2+1	5
α^9	$2x+2$	x^3+x	A
α^{10}	3	x^2+x+1	7
α^{11}	$3x$	x^3+x^2+x	E
α^{12}	$3x+1$	x^3+x^2+x+1	F
α^{13}	$2x+1$	x^3+x^2+1	D
α^{14}	$3x+3$	x^3+1	9

Tabella I

Fig. 1

Notazione Esponenziale	Notazione polinomiale binaria di $GF(4)$ su $GF(2)$	Notazione Quaternaria
0	0	0
α^0	1	1
α^1	x	2
α^2	$X+1$	3

Tabella II



p.i.: STMICROELECTRONICS S.R.L.

Fig. 2

BERGADANO MARKO
Iscritto all'Albo n. 643BI

Notazione esponenziale	Notazione polinomiale quaternaria	Notazione polinomiale binaria	Notazione esadecimale
0	0	0	0
α^0	1	1	1
α^1	x	x	2
α^2	$x+2$	x^2	4
α^3	$3x+2$	x^3	8
α^4	$x+1$	x^3+1	9
α^5	2	x^3+x+1	B
α^6	$2x$	x^3+x^2+x+1	F
α^7	$2x+3$	x^2+x+1	7
α^8	$x+3$	x^3+x^2+x	E
α^9	$2x+2$	x^2+1	5
α^{10}	3	x^3+x	A
α^{11}	$3x$	x^3+x^2+1	D
α^{12}	$3x+1$	$x+1$	3
α^{13}	$2x+1$	x^2+x	6
α^{14}	$3x+3$	x^3+x^2	C

Tabella III

Fig. 3

p.i.: STMICROELECTRONICS S.R.L.

BERGADANO MIRKO
(Iscritto all'Albo n. 843BI)

C.C.I.A.A. Torino

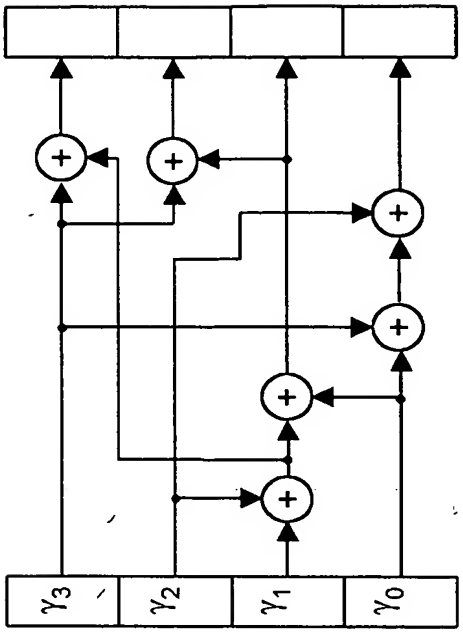


Fig.4b

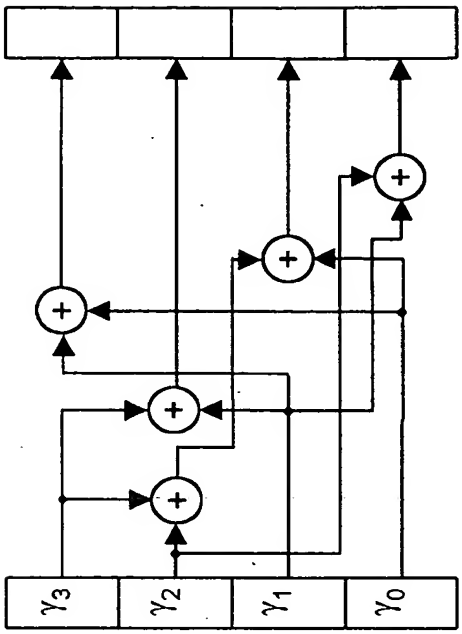


Fig.5b

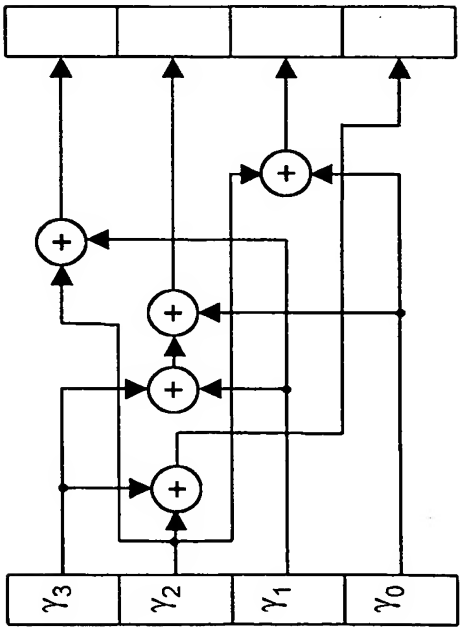


Fig.4a

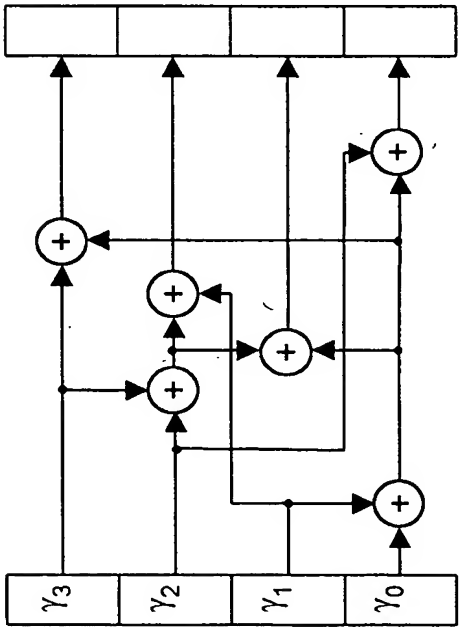


Fig.5a

p.i.: STMICROELECTRONICS S.R.L.

BERGADANO MARCO
Iscritto all'Albo n. 843BI

C.C.I.A.A.
Torino

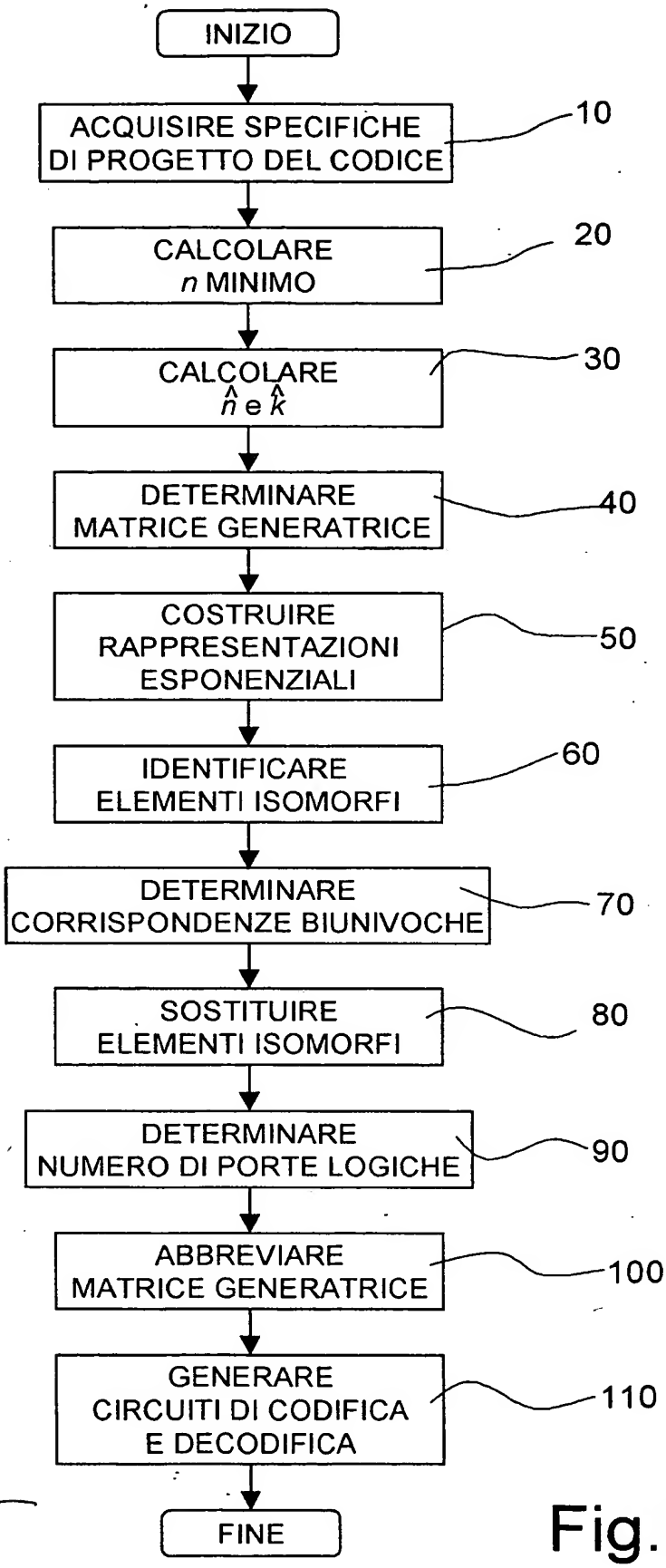


Fig.6

p.i.: STMICROELECTRONICS S.R.L.

BERGADANO MARKO
Iscritto all'Albo n. 843BM

C.C.I.A.A. Torino

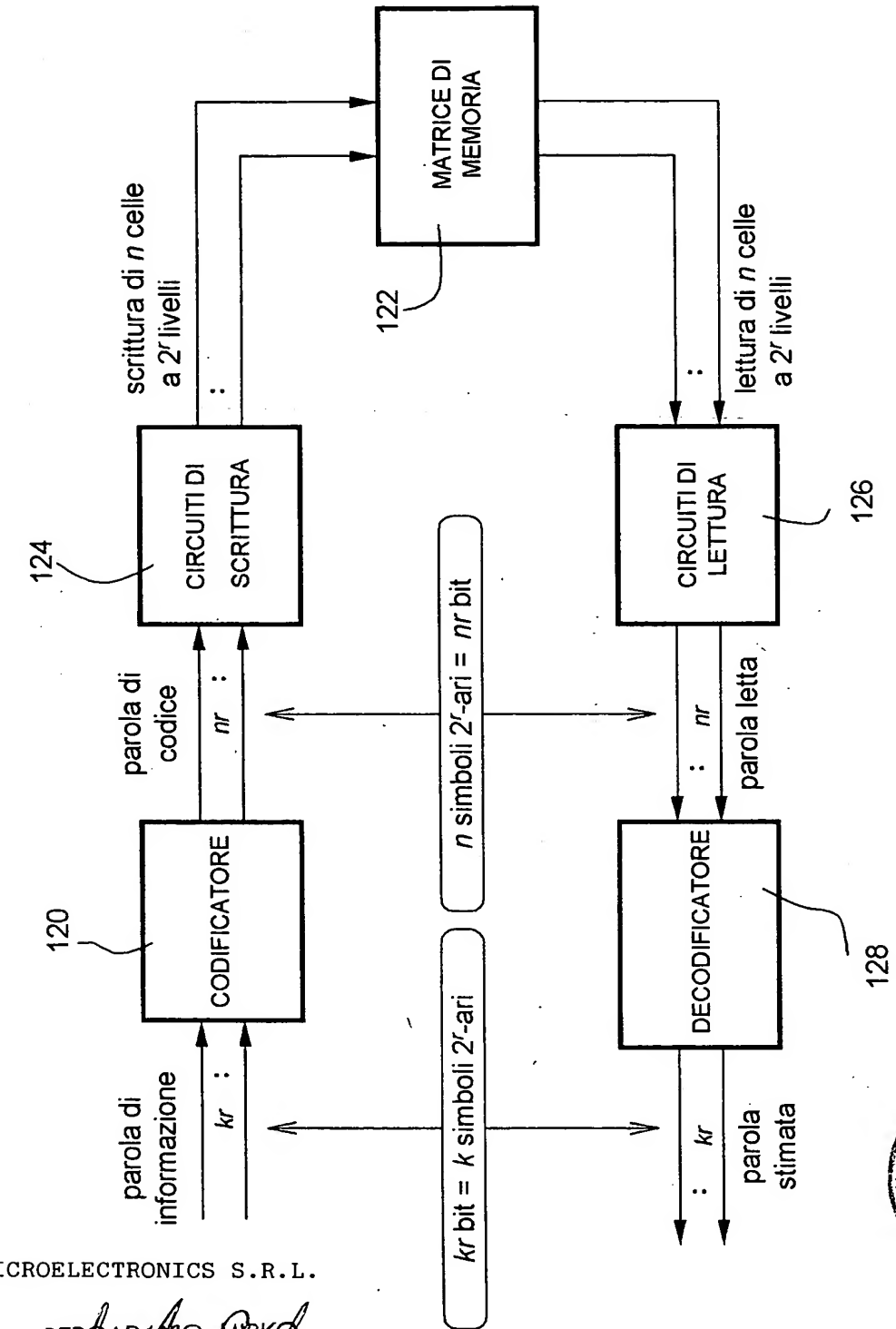
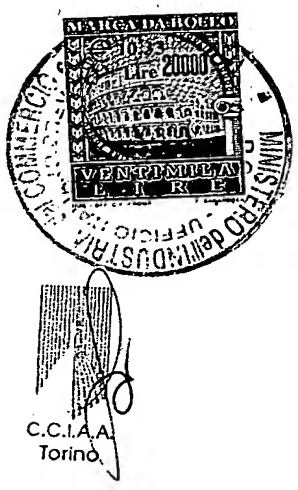


Fig. 7

p.i.: STMICROELECTRONICS S.R.L.

BERGADANO MARKO
(Iscritto all'Albo n. 548B)



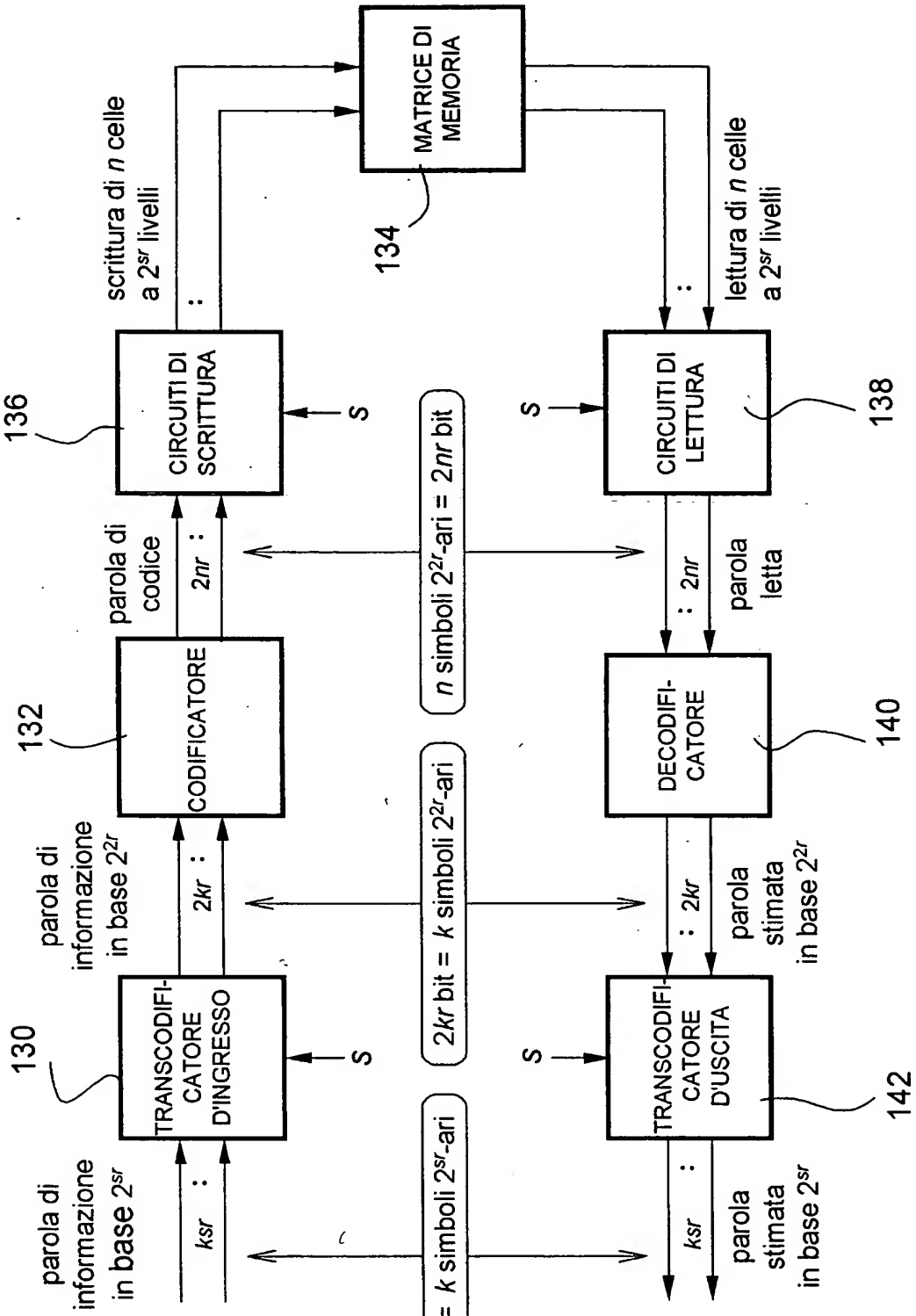


Fig.8

p.i.: STMICROELECTRONICS S.R.L.

BERGADANO MIRKO
Iscritto all'Albo n. 843BI

C.C.I.A.A. Torino